EX:No. 5 221501060

5/4/2025

**Develop a linear regression model for forecasting time series data**

**Aim:**

Write a program to implement linear regression model for forecasting time series data.

**Algorithm:**

1. **Load the Data**:
   * Read the CSV file containing the weather data.
   * Parse the date column as a datetime index.
2. **Clean the Data**:
   * Handle missing values by performing forward and backward filling.
   * Drop any remaining NaN values.
3. **Normalize the Data**:
   * Apply **Min-Max Scaling** to normalize each column's values between 0 and 1.
4. **Add Time-Based Features**:
   * Extract additional features from the datetime index: day, month and year
5. **Visualize the Data**:
   * Plot the time series for a specific column (e.g., temperature T) over time.
6. **Execute the Program**:
   * Sequentially call the functions to load, clean, normalize, add features, and visualize the data.

**Code:**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error, r2\_score

from sklearn.model\_selection import train\_test\_split

date\_rng = pd.date\_range(start='2024-01-01', end='2024-03-01', freq='H')

np.random.seed(42)

energy\_usage = np.random.normal(loc=1.5, scale=0.5, size=len(date\_rng)) + np.linspace(0, 3, len(date\_rng))

df = pd.DataFrame({'timestamp': date\_rng, 'energy\_kWh': energy\_usage})

df.set\_index('timestamp', inplace=True)

daily\_data = df.resample('D').mean()

daily\_data = daily\_data.reset\_index()

daily\_data['day\_index'] = np.arange(len(daily\_data)) # use as input for regression

X = daily\_data[['day\_index']]

y = daily\_data['energy\_kWh']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, shuffle=False, test\_size=0.2)

lr = LinearRegression()

lr.fit(X\_train, y\_train)

y\_pred = lr.predict(X\_test)

print("Linear Regression Forecasting Results:")

print("R^2 Score:", r2\_score(y\_test, y\_pred))

print("RMSE:", np.sqrt(mean\_squared\_error(y\_test, y\_pred)))

plt.figure(figsize=(12, 5))

plt.plot(y\_test.values, label='Actual Energy', marker='o')

plt.plot(y\_pred, label='Predicted Energy', marker='x')

plt.title('Actual vs Predicted Energy Consumption (Test Set)')

plt.xlabel('Time (Days)')

plt.ylabel('Energy (kWh)')

plt.legend()

plt.grid(True)

plt.tight\_layout()

plt.show()

future\_days = 10

future\_indices = np.arange(len(daily\_data), len(daily\_data) + future\_days).reshape(-1, 1)

future\_forecast = lr.predict(future\_indices)

plt.figure(figsize=(12, 5))

plt.plot(daily\_data['day\_index'], y, label='Historical Energy Usage')

plt.plot(future\_indices.flatten(), future\_forecast, label='Forecast (Next 10 Days)', marker='o', color='red')

plt.title('Energy Consumption Forecast (Linear Regression)')

plt.xlabel('Day Index')

plt.ylabel('Energy (kWh)')

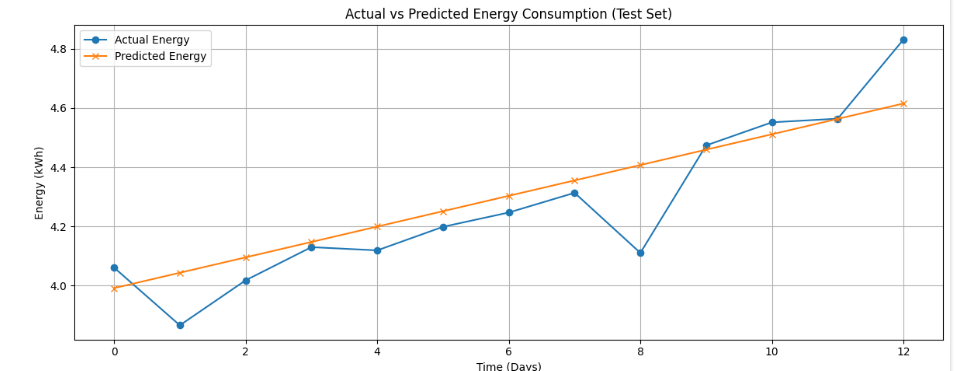
plt.legend()

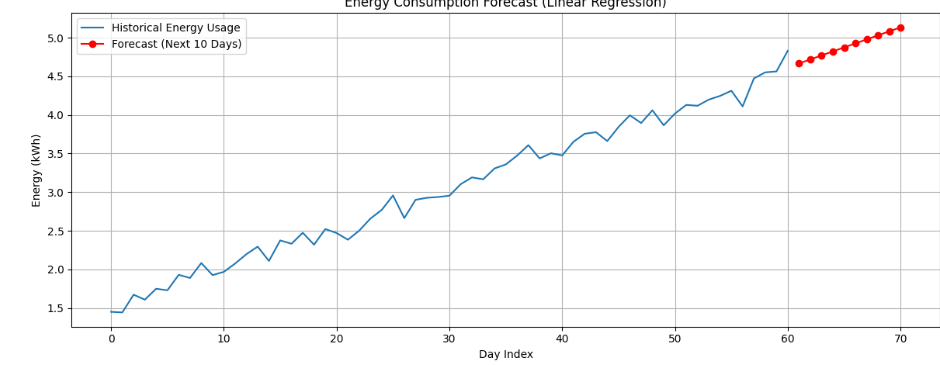
plt.grid(True)

plt.tight\_layout()

plt.show()

**Output:**





**Result:**

Thus, the program using the time series data implementation has been done successfully.